CLAIMS

What is claimed is:

1. A task management method for determining optimal placement of task components, said method comprising:

a) generating a communication graph representative of a task, task components represented as nodes of said communication graph and edges connecting ones of said nodes, said edges representing communication between connected nodes and being weighted proportional to communication between connected nodes;

b) identifying dominant edges within said communication graph;

c) determining a min cut solution for said communication graph, dominant edges being excluded from determined min cut solutions; and

d) placing task components responsive to said min cut solution.

2. A task management method as in claim 1, after the step (a) of generating a communication graph, further comprising the steps of:

a1) assigning terminal nodes, task components being placed on said terminal nodes in the task placing step (d); and

a2) identifying independent nets in said communication graph, each of said independent nets being connected between a plurality of said terminal nodes.

3. A task management method as in claim 2, wherein the step (b) of identifying dominant edges comprises the steps of:

i) identifying non-terminal nodes adjacent to at least two terminal nodes;

ii) identifying the heaviest edge among edges attached to said each identified node;

iii) identifying nodes that, when excluding the respective heaviest edge, are still adjacent to at least two or more terminal nodes;

iv) summing weights of selected edges about each of said nodes identified in step iii), excluding the respective heaviest edge, by adding the weight of the heaviest remaining terminal edge and the weights of the non-terminal edges; and

v) for each said sum, comparing the weight of the respective heaviest edge with said sum, the weight of dominant edges exceeding respective ones of the sums.

4. A task management method as in claim 3, wherein the step (c) of determining a min cut solution includes reducing the set of independent nets by selectively collapsing dominant edges responsive to said comparison.

5. A task management method as in claim 4, wherein dominant edges are selectively collapsed comprising the steps of:

i) merging nodes at opposite end of each selected heaviest edge to form a single merged node including the components of both original nodes;

ii) discarding the selected heaviest edge; and

iii) replacing groups of parallel edges with a single edge having a weight equal to the sum of parallel edge weights.

6. A task management method as in claim 5, wherein the step (c) of determining a min cut solution comprises the steps of:

i) identifying independent nets in said reduced nets;

ii) identifying and collapsing dominant edges in said identified independent nets, said independent nets being further reduced; and

iii) repeating steps (i) - (ii) until a min cut solution has been found.

7. A task management method as in claim 6, wherein each said task component is a unit of the computer program.

8. A task management method as in claim 7, wherein said each computer program unit is an instance of an object in an object oriented program

9. A task management method as in claim 7, wherein in step (d) computer program units are placed on computers, computer program units being placed on a common computer being combined into a single component.

10. A task management method as in claim 6 wherein said task is integrated circuit chip functional element placement and said task components are logic elements, said logic elements being placed on an integrated circuit chip in placement step (d).

11. A distributed processing system for determining optimal placement of computer program components on multiple computers, said distributed processing system comprising:

means for generating a communication graph of nodes interconnected by edges and representative of a computer program, computers executing said computer program being represented as terminal nodes, computer program components being represented as non-terminal nodes, said edges representing communication between connected nodes and being weighted proportional to communication between connected nodes;

means for identifying dominant edges within said communication graph;

means for determining a min cut solution for said communication graph, dominant edges being excluded from determined min cut solutions; and

means for placing program components on ones of said computers responsive to said determined min cut solution; and

said computer program being executed by said computers.

12. A distributed processing system as in claim 11, further comprising:

means for identifying independent nets connected between a plurality of said terminal nodes.

13. A distributed processing system as in claim 12, further comprising:

means for collapsing identified dominant edges.

14. A distributed processing system as in claim 13, wherein the means for identifying dominant edges comprises:

means for identifying non-terminal nodes adjacent to at least two terminal nodes;

means for identifying the heaviest edge among edges attached to said each identified node;

summing means for adding the weight of the heaviest remaining terminal edge and the weights of the non-terminal edges while excluding the identified heaviest edge; and

means for comparing the weight of the identified heaviest edge with a sum from said summing means, the weight of any identified heaviest edge exceeding said sum indicating that said identified heaviest edge is a dominant edge.

15. A distributed processing system as in claim 14, the means for collapsing dominant edges further comprising:

means for merging nodes on either end of a dominant edge and discarding said dominant edge; and

means for replacing pairs of parallel edges attached to said merged node with a single edge.

16. A distributed processing system as in claim 15, wherein each said program component is a unit of the computer program.

17. A distributed processing system as in claim 16, wherein said each program unit is an instance of an object in an object oriented program

18. A computer program product for determining optimal placement of functional components, said computer program product comprising a computer usable medium having computer readable program code thereon, said computer readable program code comprising:

computer readable program code means for generating a communication graph of nodes interconnected by edges and representative of a function, said nodes including a plurality of terminal nodes, functional components being represented as non-terminal nodes, said edges representing communication between connected nodes and being weighted proportional to communication between connected nodes;

computer readable program code means for identifying dominant edges within said communication graph;

computer readable program code means for determining a min cut solution for said communication graph, dominant edges being excluded from determined min cut solutions; and

computer readable program code means for placing functional components on said terminal nodes responsive to said determined min cut solutions.

19. A computer program product as in claim 18, further comprising:

computer readable program code means for identifying independent nets connected between a plurality of said terminal nodes.

20. A computer program product as in claim 19, further comprising:

computer readable program code means for collapsing identified dominant edges.

21. A computer program product as in claim 19, wherein the computer readable program code means for identifying dominant edges comprises:

computer readable program code means for identifying non-terminal nodes adjacent to at least two terminal nodes;

computer readable program code means for identifying the heaviest edge among edges attached to said identified node;

computer readable program code means for summing the weight of the heaviest remaining terminal edge with the weights of the non-terminal edges while excluding the identified heaviest edge; and

computer readable program code means for comparing the weight of the identified heaviest edge with the sum, the weight of any identified heaviest edge exceeding said sum indicating that said identified heaviest edge is a dominant edge.

22. A computer program product as in claim 21, the means for collapsing dominant edges further comprising:

computer readable program code means for merging nodes on either end of a dominant edge and discarding said dominant edge; and

computer readable program code means for replacing pairs of parallel edges attached to said merged node with a single edge.

23. A computer program product as in claim 22, wherein said function is a computer program and each said functional component is a unit of the computer program.

24. A computer program product as in claim 23, wherein each said program unit is an instance of an object in an object oriented program.

25. A computer program product as in claim 22 wherein said function is an integrated circuit and said functional components are logic elements.